Selection of computational environments for PSP processing on scientific gateways

Edvard Martins de Oliveira[a,*], Júlio Cézar Estrella[a], Alexandre Cláudio Botazzo Delbem[a], Luiz Henrique Nunes[a], Henrique Yoshikazu Shishido[a], Stephan Reiff-Marganiec[b]

[a] *University of São Paulo, São Carlos - Brazil*
[b] *University of Leicester, Leicester - UK*

Abstract

Science Gateways have been widely accepted as an important tool in academic research, due to their flexibility, simple use and extension. However, such systems may yield performance traps that delay work progress and cause waste of resources or generation of poor scientific results. This paper addresses an investigation on some of the failures in a Galaxy system and analyses of their impacts. The use case is based on protein structure prediction experiments performed. A novel science gateway component is proposed towards the definition of the relation between general parameters and capacity of machines. The machine-learning strategies used appoint the best machine setup in a heterogeneous environment and the results show a complete overview of Galaxy, a diverse platform organization, and the workload behaviour. A Support Vector Regression (SVR) model generated and based on a historic data-set provided an excellent learning module and proved a varied platform configuration is valuable as infrastructure in a science gateway. The results revealed the advantages of investing in local cluster infrastructures as a base for scientific experiments.

[☆] Fully documented templates are available in the elsarticle package on CTAN.
[*] Corresponding author
Email address: edvard@icmc.usp.br (Edvard Martins de Oliveira)

1. Introduction

Among the many computational systems available for scientific analyses, *in silico* experiments offer advanced results and demand higher computational capacity. Non-specialized users have shown difficulty in using complex computational solutions, due to their particularities. Therefore, simplicity of use must be one of the fundamental characteristics of such systems, once transactions and computations must be transparent for facilitating access and manipulation of the available resources.

Science Gateways consist of a set of tools, applications, and data integrated via a user-friendly portal and their main objective is to enable a larger group of users to conduct experiments, even if they are not skilled for dealing with the details of a computational resource configuration. Typically, they use tools as workflow management systems, e.g., Galaxy [1], Taverna [2], gUse [3] among others [4], [5], [6], [7], [8], for reproducibility and simplification of execution processes [9]. Such systems are integrated with databases for the acquisition of workflow steps and input data or storage of the processing results [10]. On the other hand, Science Gateways may suffer from performance traps. As they offer a high-level solution, users do not have information about servers location or installed capacity. Systems for sophisticated computations are expected to offer enough capacity, regardless of the complexity of the processing parameters, which is not always true. Hardware limitations can hamper the experiments performance, slowing the progress or delivering poor results. Protein Structure Prediction (PSP) tools, designed for discovering the native structure of a protein based on its amino acid chain [11], are an example of such complex experiments, once they demand strict controls of computer performance and generate intensive data.

Among the Science Gateways challenges are (i) selection of a gateway that is actively maintained, (ii) discovery of new services, (iii) real-time service monitoring and management, (iv) identification of sufficient computing resources for the problem complexity, and (v) support from the user's community [12]. Science

Gateways must deal with deadline constraints and paid resources for processing experiments in constant growth, therefore, the allocation of more computational resources from cloud computing is a reasonable solution. Limitations on the allocation of heterogeneous systems include different cloud providers that are not inter-operable [13] and nonexistence of a module, framework or strategy that guides the user towards the definition of an optimized computational resources configuration for running experiments.

This paper proposes a strategy based on data mining techniques for the understanding of the relationship between users input data and the behavior of the system over the execution time. The results showed Scientific Gateways require a module to help the decision on the type of resources allocation for the execution of scientific experiments. The contributions of the approach can be summarized into:

- an extensive evaluation of Galaxy capacities and shortcomings;
- a comparison of processing PSP in a varied set of machines; and
- a base module for any Science Gateway to support the computational resources identification for running experiments.

The remainder of the paper is organized as follows: Sections 2.1 and 2.2 briefly review the Service Oriented Architecture and the concepts of protein prediction structures, respectively, Section 3 introduces Galaxy framework and its mechanisms; the main proposal, materials and algorithms are presented in Section 4; Section 5 addresses the benchmark setup and the performance evaluation of executions in Galaxy environment; Section 6 reports the main works related to Science Gateways; finally, Section 7 provides the concluding remarks and suggests some future work.

2. Background

This section presents the background for this paper, which includes SOA and PSP. In both subsections are included some references of each research field.

2.1. Service Oriented Architecture

Research institutions routinely deal with various types of data and coding systems, which are fragmented and spread among data repositories and lead to information loss and increased costs of their systems [14]. Cross-platform integration can be provided through a Service-oriented architecture (SOA)-based solutions that support scalability, re-usability, and heterogeneity [15]. Therefore, web services integrate different repositories and platforms while maintaining their privacy policies and adaptability [14].

A SOA-based system is comprised of the following three main elements: (i) service provider, which hosts independent web services to perform procedures; (ii) service repository, which stores the description of services, location and accessibility; and (iii) client application, which requests services to a host based on its service description [16].

Services can be integrated and distributed by public clouds, dedicated clusters or multi-architecture systems with processing GPUs. Besides those dynamic environments delivering high performance computing the requirements for a dependable architecture involve heterogeneous resources, scalability and minimization of communication, among others [17] [18] [19].

Abdul-Wahid et. al. [20] proposed a framework based on Work Queue (WQ) and python implementation that provides scalability and integrates clusters with cloud solutions for concurrent processing. Pronk et. al. [21] worked with three levels of parallelism (SIMD, threads, and message-passing) and provided automatic resource allocation. Such studies show the complexity of integrating resources and technology for research and the ways computation can act.

2.2. Protein Prediction Structures

This section describes the protein chains and the PSP processes for a better understanding of their importance and use as workload. Proteins are linear polymer chains that perform fundamental biological functions for the maintenance of life [22], once they regulate most activities within live organisms, as replication of genetic code and maintenance of cell shape [23]. Proteins are formed from amino-acid sequences (also known as residues) linked by peptide bonds. Two amino acids form a peptide bond when they concatenate by releasing water. The folding process aims at defining the native state of a protein from an inactive chain [23]. Scientists have focused mainly on two techniques to define those structures, namely X-ray crystallography and nuclear magnetic resonance (NMR). Although precise, the technologies are costly and the processes are slow, which have motivated the research on PSP, which aims at predicting the tertiary structure of a protein based on the amino-acids chain.

The folding process is a physical-chemical method in which any of the strands displays a functional configuration for a specific 3D structure [24]. Four basic representations of proteins are defined according to the residues arrange: the primary structure is the representation of the amino-acid chain; the secondary represents the hydrogen bonds and a basic order; the tertiary structure is the three-dimensional (or native) form, which enables the protein to perform biological functions. It is energetically stable and can be used in drug design or for the understanding of behavior of diseases. Finally, the quaternary structure is a representation of multi-tertiary complexes. Figure 1 shows the four structures [24].

The amino-acid chain is informed as input for the PSP processing phase. In the sequence of the execution, many structures are produced until the best one has been achieved, which represents the most stable one. The identification is defined by the structure of lowest energy and can be expressed by the following formula: if $c \in C(s)$ where $c(s)$ represents the family of valid sequences for a given set and $E(c)$ is the energy function of the desired structure, the general formula is defined as min=$\{E(c)|c \in C(s)\}$ [25].

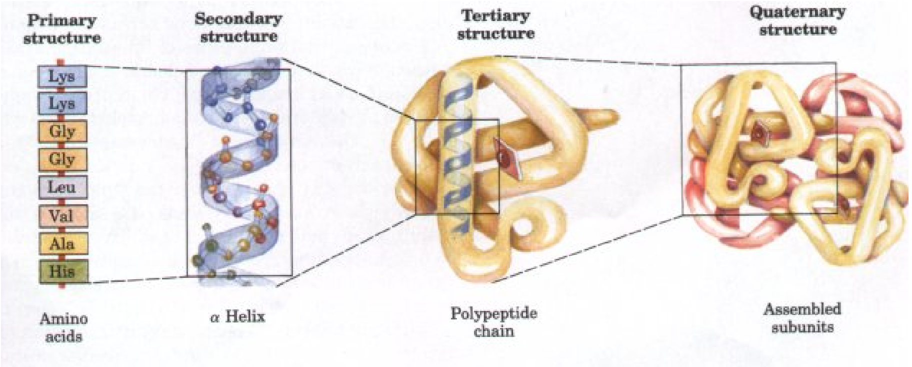


Figure 1: Four types of protein structures [24].

Research on prediction of protein structures has faced obstacles, despite the evolution of technology [25]. The main objective of researchers is to improve the prediction quality, however, they rarely focus on computational efficiency or reduction of processing time [26].

The impact of similar research on public health is an example of the importance of the field. Over a decade ago, Amato et. al. [27] revealed scientists resorted to distributed systems, which are currently much more advanced. Other approaches are presented in [28], [29] and [30]. This field is close to health-related research, as in [31], [32], [33] and [34]. Reduced prediction error is presented in [35], and improvements in information on protein structures can be found in [36], [37] and [38]. Finally [39] describes the I-Tasser framework hosted on servers exclusively for PSP. The majority of improvements are protein-related methods.

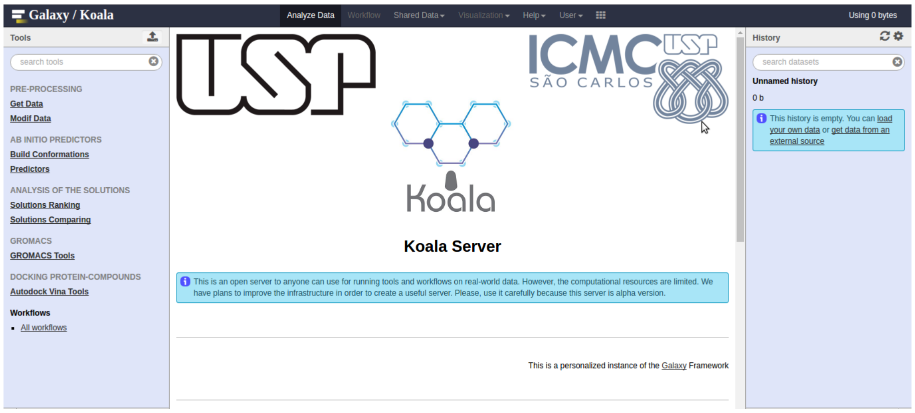
Many of the above-mentioned studies suffer from performance and computational provision problems. Cloud Computing is the preferred platform, once it is powerful; however, it implies cost limitations and third-part management. The integration of clusters and other platforms can be an important advance for ensuring higher computational capacity and reductions in the processing time. Some of the studies presented must be operated by scientists with little experience in complex systems. As Science Gateways are systems with tools for processing and data analysis while maintaining transparency, they simplify the access to users. As the PSP experiments are computationally consuming, they require a careful infrastructure definition and help for the identification of limitations of those gateways.

3. Materials

Galaxy is an open-source and modifiable web platform widely used in research on bioinformatics. Servers are available for free and users access analysis tools and mechanisms for running and reproducing workflows. Some algorithms, methods and analysis tools can be integrated in the framework, which makes it personalized and adequate to users' needs [40]. The main goals of Galaxy are (i) access to complex computational resources from a broad public, (ii) reproducibility of experiments and (iii) collaborative analyses via web [41].

Galaxy has been built for an easy access by unskilled users. It has a general purpose and is used in several domains with the same quality. Some of its principal features include accessibility, reproducibility and transparency. The Galaxy Project is divided into the following parts:

- Galaxy Server: The Galaxy project[1] provides computational resources to Bioinformatic experiments. Public servers can be found in other research pages.



Koala Server

This is an open server to anyone can use for running tools and workflows on real-world data. However, the computational resources are limited. We have plans to improve the infrastructure in order to create a useful server. Please, use it carefully because this server is alpha version.

This is a personalized instance of the Galaxy Framework

This history is empty. You can load your own data or get data from an external source

Figure 2: Example of a Galaxy implementation available for use at ICMC servers.

- Galaxy software framework: an open source project that offers customization and integration to provide services. This research has been con-

[1] https://usegalaxy.org

ducted in an instance developed at the Institute of Mathematical Sciences and Computation (ICMC)[2] and named Koala[3], which provides data management, PSP predictors, analysis tools and workflow assistance. The interface is shown in Figure 2 [42].

- Galaxy Tool Shed: space for the sharing of tools, solutions and steps for configuration and installation. It is available at the project page [4].
- Galaxy Community: the main part of the project, it is composed of developers, users and administrators that work together for evolution and updates. Discussions and collaboration are primordial for any open-source endeavor [40].

In comparison to other platforms, Galaxy stands out because of its many resources and great community [40]. It can be used via web interface, regardless of its installation and is a considerably easy system for manipulation. Figure 3 shows the process of use of Galaxy Koala architecture. The user inputs the data, usually collected from a public storage, as PDB. They are informed to the framework via direct download or upload tools. The user chooses the algorithms that process the data according to the experiments design. If necessary, intermediate data (i.e. population files of a genetic algorithm) are produced. The processing phase starts, the results are generated, and the user decides on storing or moving the data.

Despite its penetration in scientific fields and its many tools that help research conduction, several aspects of Galaxy (and also Koala) have not been optimized. Some important limitations regard monitoring of active processes, difficulties with large data-sets and lack of information about machine hosts. The lack of information on computational capacities limits the design of experiments and can cause failures that cannot be avoided by users. The system

[2] http://www.icmc.usp.br

[3] http://koala.lasdpc.icmc.usp.br

[4] https://usegalaxy.org/toolshed

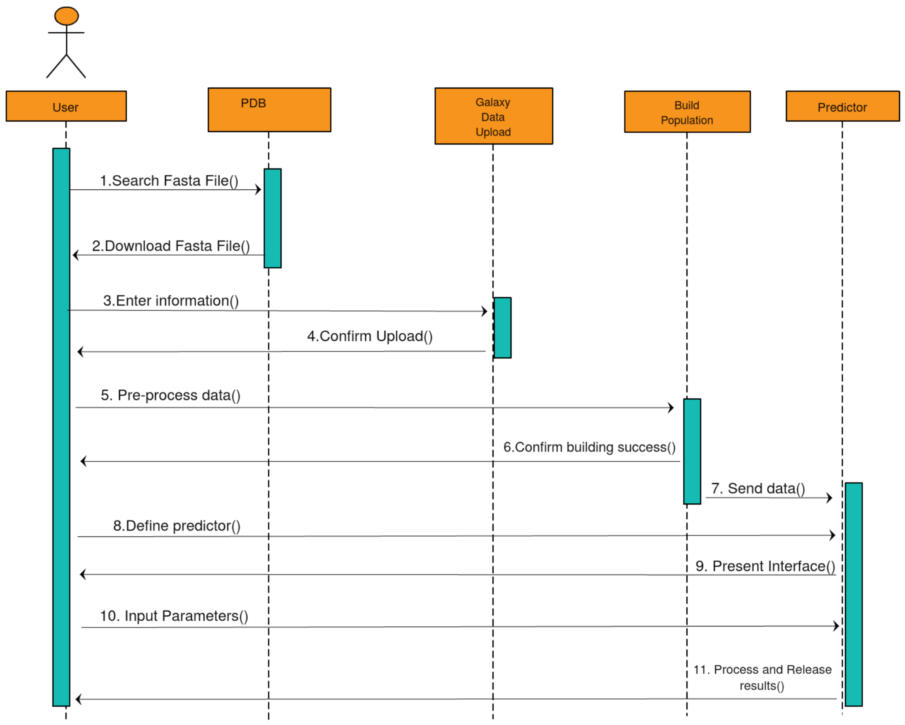


Figure 3: Sequence diagram of an experiment in Galaxy Koala.

cannot detect or inform on the problem, and in an attempt to finish an execution, it can cause a dead lock. The user receives zero notification about the system or the experiment status and waits for a conclusion that may not be reached. The access control is also a problem, once only the administrator can check and fix most faults. Galaxy has been designed for a broad public, including non-skilled users, which is a major drawback in its usability. Once Galaxy Koala' hosts cannot be chosen, the user must rely on the instances offered.

A survey into Galaxy Koala components, types of faults and extent of damage based our previous studies and the literature review is shown in Table 1. The components may be tools offered by Galaxy that report mechanisms or hardware devices. Predictors are tools developed for PSP experiments, providers nodes are the hosts that offer the services and feedback is the mechanism that informs users on the conclusion of the experiment. Detectability describes the system

Table 1: Galaxy Koala components and most common failures.

Component	**Failure**	**Domain**	**Consequences**	**Detectability**	**Frequency**
Predictor	Crash	Content	Medium	Yes	Occasionally
Predictor	Input	Content	Minor	Yes	Often
Memory	Insufficiency	Time	Major	Yes	Often
HD	Insufficiency	Content	Fatal	No	Rare
Feedback	Crash	Time	Minor	No	Often
Monitoring	Nonexistence	Content/Time	Major	No	Often
Provider Node	Crash	Content	Major	Yes	Occasionally

operation and frequency indicates how often faults occur. Such shortcomings indicate the improvements required and that the maintenance of Science Gateways must be constant. Users' needs, technologies available, complexity involved and possible gains must be accordingly evaluated. Problems similar to the ones in Koala are found in other Galaxy implementations [1].

4. Methodology

Critical points of Galaxy Koala were identified in our assessments. The modules common to Galaxy and other Science Gateways can be defined after a careful study, as shown in Figure 4. The portals are accessible through an interface, usually a web page, for authentication and definition of access levels. Workflow managers define the sequence of tasks execution, as well as data management. Management tools, such as schedulers, monitors, and load balance comprise the next module. Finally, the execution module forwards the requests to machines and repositories for execution, storage and retrieval of information. The extra module in this infrastructure is the *Decision Maker* proposed here. This component performs a previous analysis of the user's needs and indicates the computational configuration for the execution of an experiment with best cost-benefit. It differentiates the best environment for a request and reduces costs of cloud machines and network traffic.

Decision Maker uses machine learning to process past runs, discover patterns on a system's behavior and define the best configuration for each data entry. It uses this information to toggle the sending of requests between infrastructures installed locally in desktop machines, computer clusters, or cloud instances. According to the parameters, it chooses between running on local servers with downloaded information or sending the data to remote servers.

4.1. Environment selection

On many occasions, defining the configuration for a processing environment towards obtaining the best performance is a hard task. A machine of high capacity can probably process many experiments, but also wastes electric power and computational resources when dealing with smaller tasks. Modest resources, on the other hand, may cause delays, wrong answers or process starvation. Another problem is related to inflexibility in classic workstations, limited by hosts' capacity and difficult update. Clusters combine the capacity of various machines and are a good option to offer high-demand resources. They are also limited

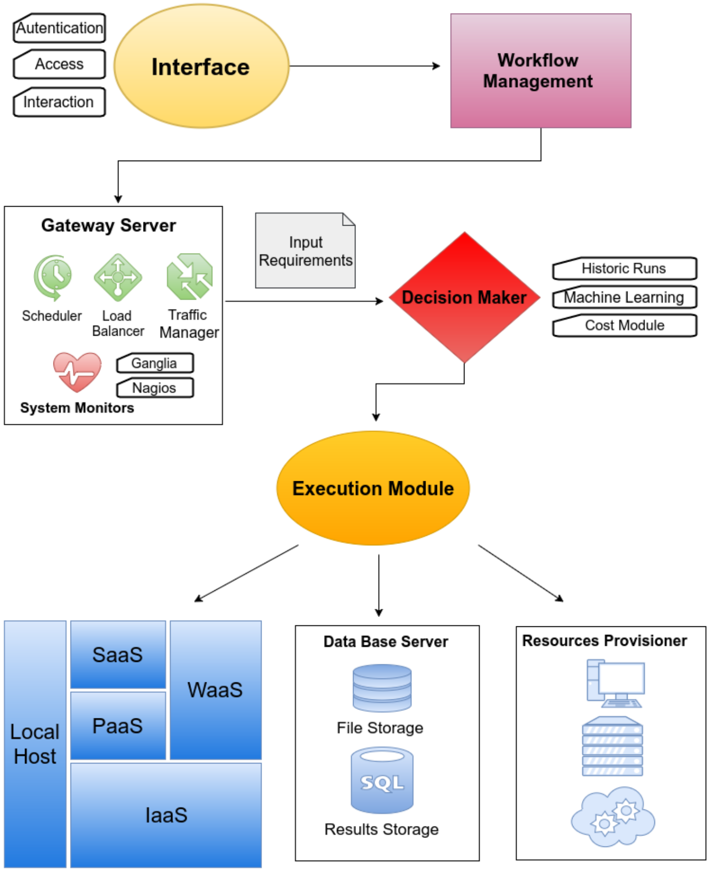


Figure 4: Basic common modules of procedural architectures. The Decision Maker module is a gap in this technology that can help the creation of scientific processes with good performance.

by the number of machines available and create parallel computing complexity to programmers. The other option is cloud computing, which offers elasticity, pay-as-you-go models and availability in a range of prices and according to the investment. If researchers have to deal with all such options during experiments, the quality is reduced and errors can occur.

Decision Maker makes decisions over the configuration of the machines that will execute the experiments. The users' parameters combined with previous experiments are used in evaluations for the definition of the best solution to each case. The outputs of this module are the best suitable configuration of an environment such as a workstation, a VM on cluster or cloud instance. This study aims at contributing to research on PSP providing an environment in which experiments yield results in an optimized way, by either cheaper and/or faster methods. Therefore, Quality of Service (QoS) metrics, usual in SOA, must be respected and specific definitions for the research field must be provided. This article proposes an infrastructure based on multi-environments, selection of the best configuration for each set of experiments and supply of resources in an integrated form. Galaxy Koala and its tools can be offered through a diverse set of machines that work either locally or via the Internet. The following three setups were defined and classified by the computational capacity: a desktop machine, a cluster machine and a cloud machine. The execution of a given experiment in the proper machine can avoid delays and waste of resources. Figure 5 shows the heterogeneous environment in a crescent order.

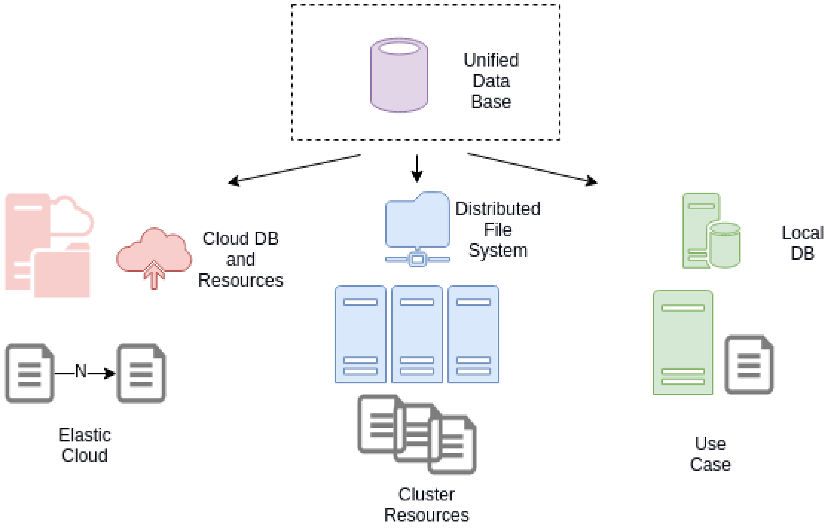


Figure 5: Environments considered for the architecture.

4.1.1. Computational Capacity

Computational resources may vary regarding capacity. The performance of desktops has significantly improved, however, the two main resources for scientific purposes are clusters and cloud computing. Each of them includes advantages and shortcomings for maintaining Service Level Agreements (SLA) [16]. Galaxy Koala can be installed in any of such resources, and the choice will vary according to the initial specification and experimental design. When running in a private cluster, the administrator can choose among many aspects for configuring an environment and deciding on the steps of an experiment. Working locally is a good choice for researchers who have expensive computers at hand and are technically skilled to manage the system.

Clusters are a congregation of machines prepared for parallel computing and to offer combined capacity. They are usually available in companies and universities and some of them offer public access. They are a very good option for running experiments, and, in most cases, although the final user does not need to deal with the maintenance of hardware and software, the administrator's privileges are lost.

Finally, cloud computing provides resources on demand. It could be the solution to every capacity problem, however, costs and bandwidth are some of its limitations. The Internet speed has not grown proportionally to the size of data generated, which represents a prohibitive aspect in many cases. When running experiments exclusively on the cloud the ideal is to allocate virtual machines closer to the database, for the avoidance of data transportation and storage the results in disperse repositories. Such aspects brings new concerns regarding the maintenance and privacy of information.

To work as the computational environment, a diverse set was defined as follows: the working machines were divided into three different classes, namely "desktop", "cluster" and "cloud" and their different configurations are shown in Table 2.

4.2. Algorithms

This section introduces the machine learning algorithms used for learning from data obtained by previous protein experiments in the Galaxy Koala platform. Each data point is referred to as vector $(\mathbf{x}, y)$, in which $\mathbf{x} = (x_1, x_2, \ldots, x_d)$

Table 2: Three Galaxy Instances

Environment	**Hard Drive**	**CPU**	**RAM**	**Operational S.**
Desktop	20 GB	2 Core	8 GB	Linux Ubuntu 14.04.4 LTS
Cluster Node	50 GB	4 Core	16 GB	
Cloud VM	200 Gb	8 Core	32 GB	

is a d-dimensional vector of *input variables* (or features) and y is a single output variable, i.e., the *target variable.* The objective of a learning algorithm is to produce a *predictor* $\varphi()$, trained by the learning set $\mathcal{L} = (\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)$. If the produced $\varphi()$ is a good predictor, i. e., if it has learned from data, it can predict unseen data with a small error, as long as this data point has been sampled from the same distribution of $\mathcal{L}$.

As the target variable y_i of our experiments is a numerical value (the processing time), algorithms referred to as *prediction modeling* were chosen. Three different algorithms of linear regression, and one based on Support Vector Machines (SVMs) were assessed. Different algorithms were chosen for the task because different machine learning algorithms learn differently from data, i. e., depending on the data, some learning algorithms are better to learn than others.

The *linear regression* method approximates a formula as the one shown in Equation 1, in which $W = w_0, w_1, w_2, \ldots, w_d$ is the vector of coefficients. Therefore, the obtained predictor is a dot product between input variables $X = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)$ and coefficients W estimated by the minimization of squared error, shown in Equation 2.

$$y_i \approx w_0 + w_1 \cdot x_{i,1} + w_2 \cdot x_{i,2} + \ldots + w_d \cdot x_{i,d}, \quad \text{in which } i \in 1, 2, \ldots, n \tag{1}$$

$$O = \sum_{i=1}^{n} (W \cdot X_i - y_i)^2, \text{ in which } i \in 1, 2, \ldots, n \tag{2}$$

Smaller coefficients W are desirable, because they reduce *overfitting* [43]. The two approaches for linear regression that are more effective for such reductions are *ridge regression* and Lasso. In Ridge, an L_2-penalty term is summed to objective function O of Equation 2. Such as a $\lambda||W||^2$, where $\lambda > 0$, penalizes the linear regressions with larger coefficients. On the other hand, Lasso algorithm uses an L_1-penalty summed to O, $\lambda \sum_{i=1}^{d} |w_i|$, and obtains sparse solutions, i.e., solutions with few nonzero components. Such a characteristic is specially effective when irrelevant features are present.

SVMs were initially developed for the binary classification of numeric data. They focus on the optimization of hyper-plane that maximizes the margin between the two classes, which is proven to generalize well to unseen data as it is grounded in the framework of statistical learning theory [43].

An SVM can be used as regression using the method ϵ-SV regression [44]. Its goal is to find a linear function $f(x) = W \cdot X_i + b$, with $w \in \mathbb{R}$, and $b \in \mathbb{R}$, that has at most ϵ deviation from the actually obtained targets y_i for all training data, i. e., if the prediction error is lower than ϵ, it is not considered. At the same time, it seeks small w, a property called flatness. The advantage of the use of SVMs is they can be coupled to kernels, thus transforming the input data into another, higher dimensional space, which enables the separation of nonlinear data.

Table 3: Andromeda Cluster Details

Cluster Andromeda	
Number of Hosts	13
Processor	AMD Processor Vishera 4.2 Ghz
RAM	32 GB RAM DDR3 Corsair Vegeance
Hard Drives	HD 2TB Seagate Sata III 7200RPM
GPU	Video Nvidia GTX 650 1GB
Motherboard	Gigabyte 970A-D3
Power Supply	ATX 650W Real
Operational System	Linux Ubuntu 14.04.4 LTS

5. Results and Discussion

An elaborated scenario was prepared on the Galaxy Koala Framework for the collection of results and modeling of data behavior through machine learning, so that the best computational environment could be properly defined. Experiments were performed at cluster Andromeda available at the Distributed Systems and Concurrent Programming Laboratory (LaSDPC)[5] (see details in Table 3).

Proteins in the last Critical Assessment of protein Structure Prediction (CASP), hosted in 2016[6], were studied for the construction of a strong background. CASP is one of most important events regarding PSP and the data used by the groups involved are a good point of investigation. Below are two histograms with the protein sizes considered Figure 6 shows the target proteins used as goals and disputed by the groups with the best predictions. Figure 7 displays the Domain Definition proteins and their sizes exhibit a similar frequency.

A diverse set of parameters was defined for the experiments, also varying the

[5] http://infra.lasdpc.icmc.usp.br/

[6] http://predictioncenter.org/

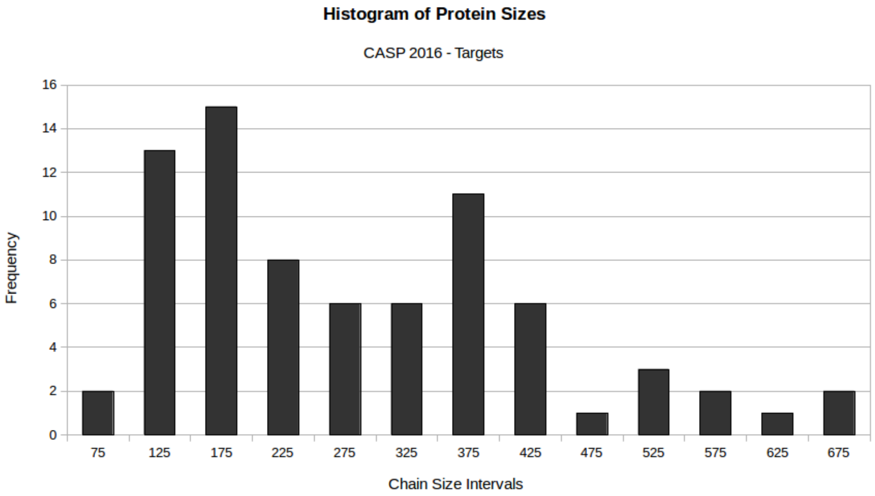


Figure 6: Histogram of the proteins size processed at the last CASP, in 2016. Those are the proteins targets.

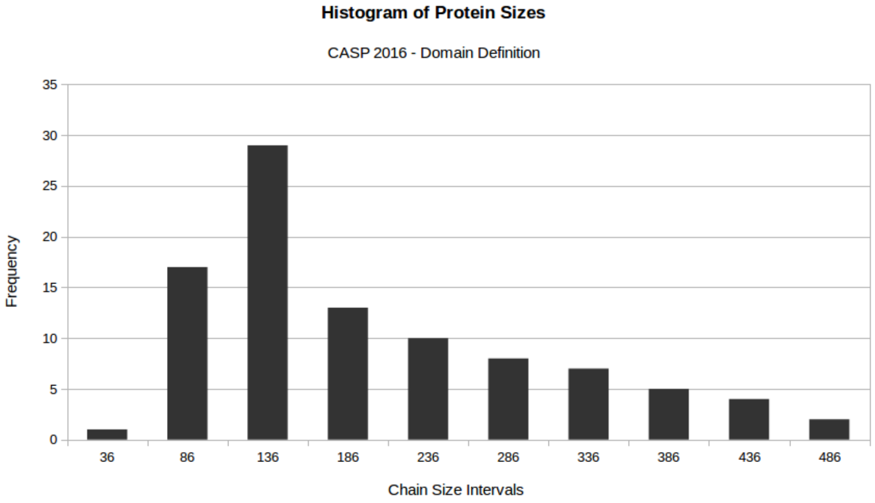


Figure 7: Histogram of the proteins size processed at the last CASP, in 2016. Those are the domain definitions.

protein chain sizes. 2PG Mono [42] is the predictor chosen that uses a genetic algorithm to define the protein objective form. Besides protein size, the two

Table 4: Proteins and parameters for the experiments.

Protein	**Chain Size**	**Population**	**Generations**
2N0L	12	From 20 to 30	From 10 to 150
1BCV	20	From 20 to 30	From 10 to 150
1AI0	21	From 20 to 500	From 20 to 1000
2ETI	28	From 20 to 200	From 20 to 400
1C94	38	From 20 to 200	From 20 to 1000
5K2L	49	From 20 to 200	From 40 to 800
5HVZ	50	From 20 to 100	From 10 to 400
1B1G	75	From 20 to 50	From 40 to 400
1EOD	100	From 100 to 300	From 200 to 1200
1CM7	100	From 30 to 100	From 30 to 100
1OZ9	150	From 20 to 50	From 30 to 200
4IEU	200	From 50	From 2 to 30
1AGY	200	From 20 to 50	From 30 to 300
2EEK	220	From 20 to 30	From 20 to 200
2R3A	300	From 20	From 60 to 200
2BX6	350	From 20 to 50	From 40 to 150

other parameters are population size and number of generations for the genetic algorithm. They were chosen without a pattern towards making their prediction harder. Table 4 shows the respective values.

In a first approach, the experiment was conducted in a homogeneous set of machines for a better understanding of the behaviour of the workload and the differences imposed by the change in the parameters. The results are discussed in the following charts. Figure 8 shows the overall processing time for all proteins. The curve displays a growing tendency, according to the parameters input. The higher parameters in the last columns show the impact of the larger chains on the system. Proteins of this size are common in many PSP experiments, therefore, systems should be prepared for them.

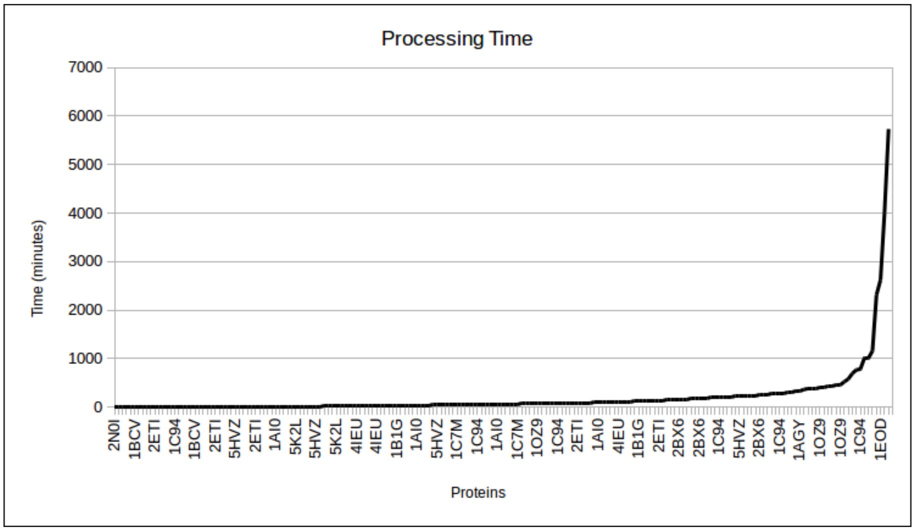


Figure 8: Processing time results for all scenarios.

For a better perspective on processing time growth, protein 1EOD is not included in Figure 9, because the load it imposes on the system distorts the scale. The chart shows the differences in the processing time of the proteins. Some may appear more than once in the X axis, as the parameters of population and generation change.

Figure 10 displays the largest processing time for each protein. The workload overhead imposed on the infrastructure ranges from minutes to several hours. Even smaller initial chain sizes may be in the process for long times, i. e., up to days.

Three proteins, namely 2N0L (12 amino-acids), 1B1G (75 amino-acids) and 1AGY (200 amino-acids), were chosen for a comparison of samples according to their sizes. The aim was to present not only the differences between the input parameters, but also the way machines of lower capacity can deliver good results if the workload is suitable. The differences are shown Figure 11.

The information enabled some data evaluations for the definition of a general model that correctly represents the relations among experimental parameters, machine capacities and response time. As discussed in sections 4 and 5, lower-

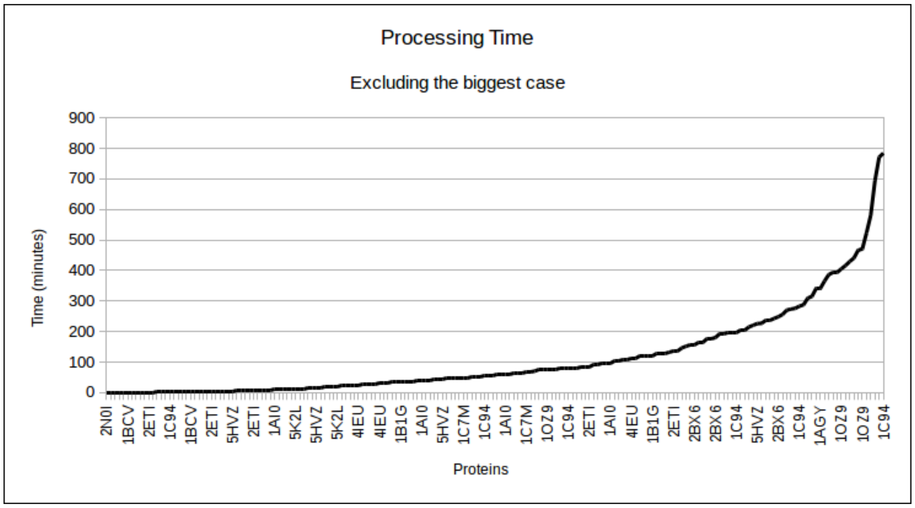


Figure 9: Processing time results after the exclusion of protein 1EOD.

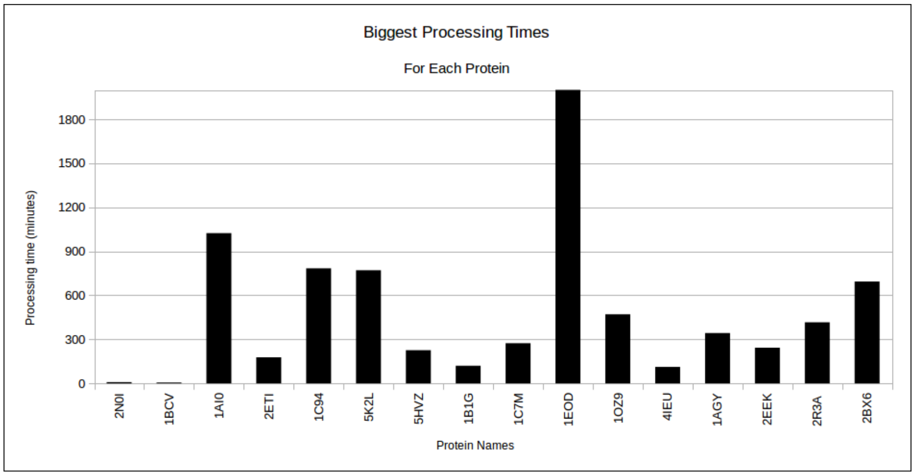


Figure 10: Larger processing times recorded.

capacity machines fulfill users' requirements. Once Galaxy is adaptable to a variety of computational solutions the working machines were divided into three different classes, namely desktop computer, cluster machine and private cloud inside LASDPC. The different configurations for the machines are shown in Table 2, Section 4.2. The same experimental parameters of Table 4 were used, but in this case, different machine classes were chosen according to the input.

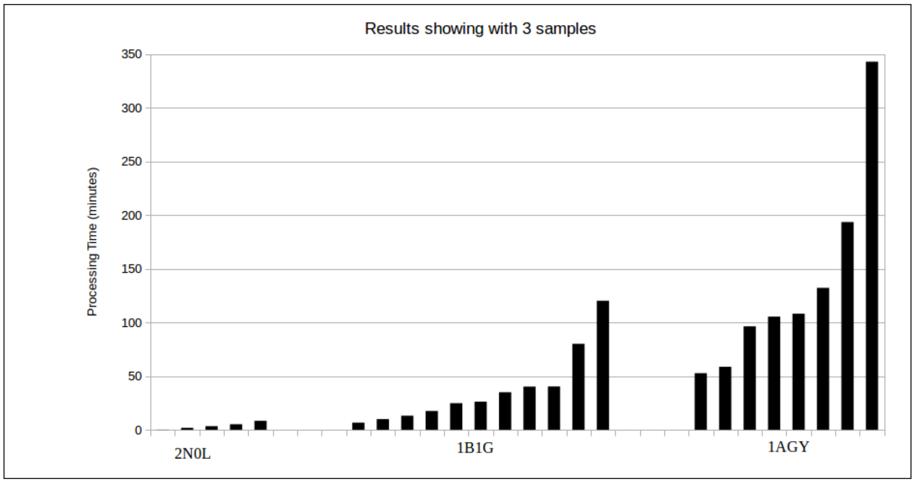


Figure 11: Sample of three PSP calculations. The results may overlap the higher classes, which hampers the prediction of the system's service capacity.

The experiments were conducted in Koala, as reported in Section 4, and created a database of historic executions used for the regression models. The algorithms chosen for the regression are discussed in Section 4.2.

First, the data were loaded and a target variable was defined. Once, in this case, "Processing time" was the target, the best time for the execution of a protein in a selected machine could be estimated. Some information, such as protein names and resulting file sizes were discarded from the database because the regression tests did not use it. The data were then divided into two sets, namely train and test [45].

A few trials were necessary for achieving the modeling that suited the type of data collected. Three linear regression techniques were experimented and unsatisfactory results were yielded. Each model was tested with four types of data transformation, namely raw data, normalized data, log and normalized log. A few transformations in the raw variables, as conversion of strings to numerical values and scaling of numerical variables to mean=0 and standard deviation=1, were required [45].

The first experiment was the basic Linear Regression, however, the error

Table 5: General results for the linear models over the collected data. Labels: LR (Linear Regression), LL (Lasso Linear) and RR (Ridge Regression). R Squared: Ideal is closest to 1. RMSE(Root Mean Squared Error): The less the better.

Model	**Score Function**	**Raw**	**Normalized**	**Log**	**Normalized Log**
LR	R Squared	0.48	0.48	0.53	0.53
LR	RMSE	932.72	932.72	881.05	881.05
LL	R Squared	0.48	0.49	0.53	0.53
LL	RMSE	930.74	920.41	880.91	880.32
RR	R Squared	0.48	0.49	0.54	0.54
RR	RMSE	931.53	924.74	875.63	875.44

rate was unsatisfactory [46]. Least Absolute Shrinkage and Selection Operator (Lasso) Regression were then tested, with L1 penalty, however, again, the results did not fulfill the requirements for a good model [47]. Ridge Regression with L2 Penalty [48] was the third regression to be tested and also yielded inadequate results. The general results for the models are shown in Table 5.

A non-linear model was applied to the data and results analyzed. Support Vector Regression (SVR) is a variation of Support Vector Machines (SVM) used for classification, regression models and construction of strong models. SVMs were initially developed for the binary classification of numeric data. They focus on the optimization of the hyper-plane that maximizes the margin between the two classes, which is proven to generalize well to unseen data as it is made in the framework of statistical learning theory [43]. SVR was implemented with the train set and its performance was assessed by predicting never seen data using the same data set. The following score functions were defined for the modeling of SVR:

1. R^2 (R-squared): as close to 1 the output, the better.
2. RMSE (Root Mean Squared Error): the lesser, the better.

The overall results were superior than those of linear regressions. As shown in Table 6, SVR achieved R squared results close to the ideal ones and different RMSE numbers. The best results were provided by the normalized data, with a small deviation.

Table 6: Results of the running of SVR model on the collected data. Best results highlighted.

Model	Score	Raw	Normalized	Log	Normalized Log
SVR	R Squared	0.98	**0.997020**	0.94	0.95
SVR	RMSE	174.64	**70.505679**	312.44	293.05

Figure 12 displays the predicted values obtained by the SVR plotted versus true values. The points follow a diagonal line, with few variability, which has proven that the predictions are very close to the expected values.

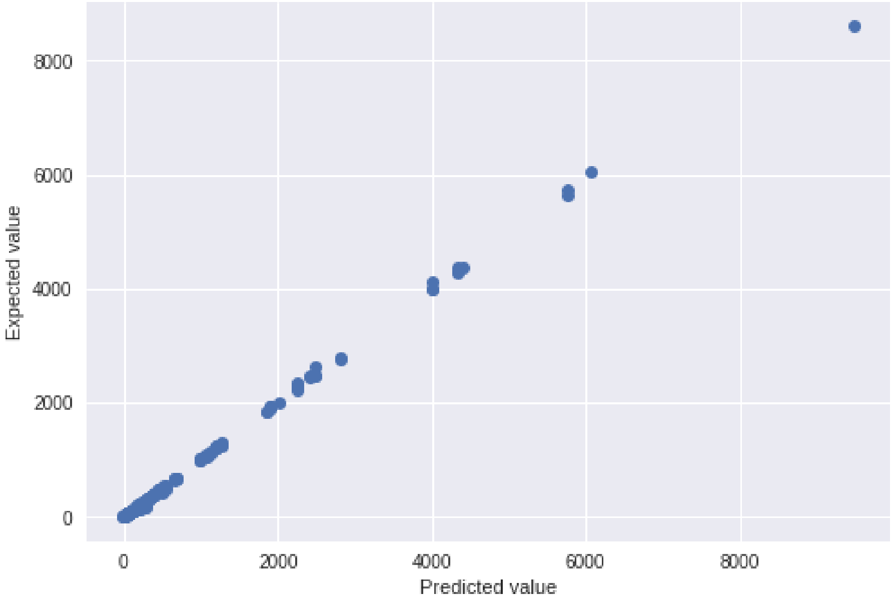


Figure 12: Results from SVR learning algorithm. The accuracy is high, with few deviations.

The model generated is now feasible to try the SVR with information not available in the data-set and have a prediction of the processing time for each environment. By adding the parameters as any set of the experiment and informing the three types of machine available, the model estimates the processing time for each one and indicates the best option. The three environments were tested with the same parameters below:

- Protein chain size = 72 amino-acids

- Population size = 80 individuals
- Generations = 200

Figure 13 shows the results.The desktop environment displayed the worst results, while cluster and cloud machines exhibited similar response times. The cluster machine provided a slightly better result than the cloud, which has proven local environments can properly attend users in some cases. Even in a tied result, the use of the cloud must be avoided, due to bandwidth and costs involved.

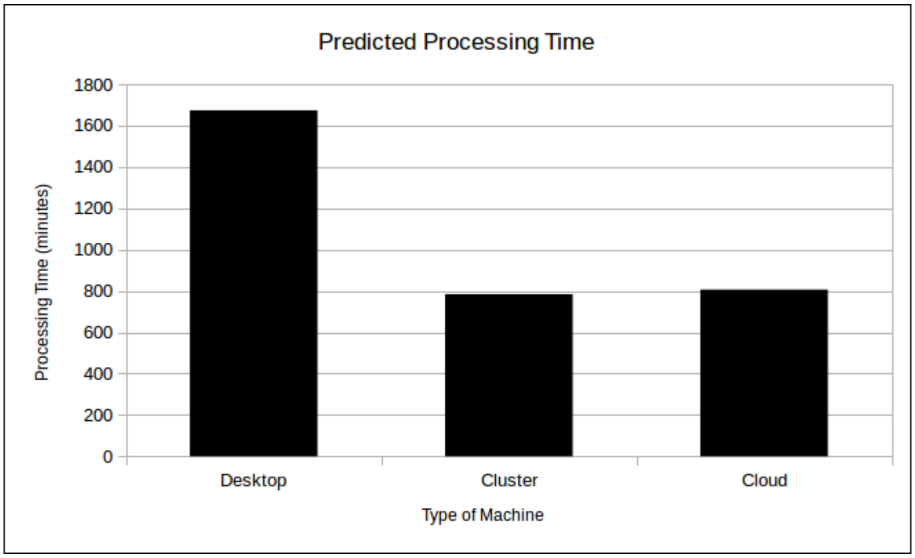


Figure 13: Results predicted by the SVR model on never seen data. The best result in this sample was achieved by the cluster machine.

The model predicts the ideal environment for each PSP experiment defined in the scenario and Decision Maker can correctly determine the workload that fits a machine configuration. Considering the complete data-set, the best type of machine according to the workload is observed in the Figure 14. The almost equal job distribution and the less powerful machine (desktop) chosen in 41% of the experiments show the necessity of an evaluation of each case for the avoidance of waste of resources and computing capacity.

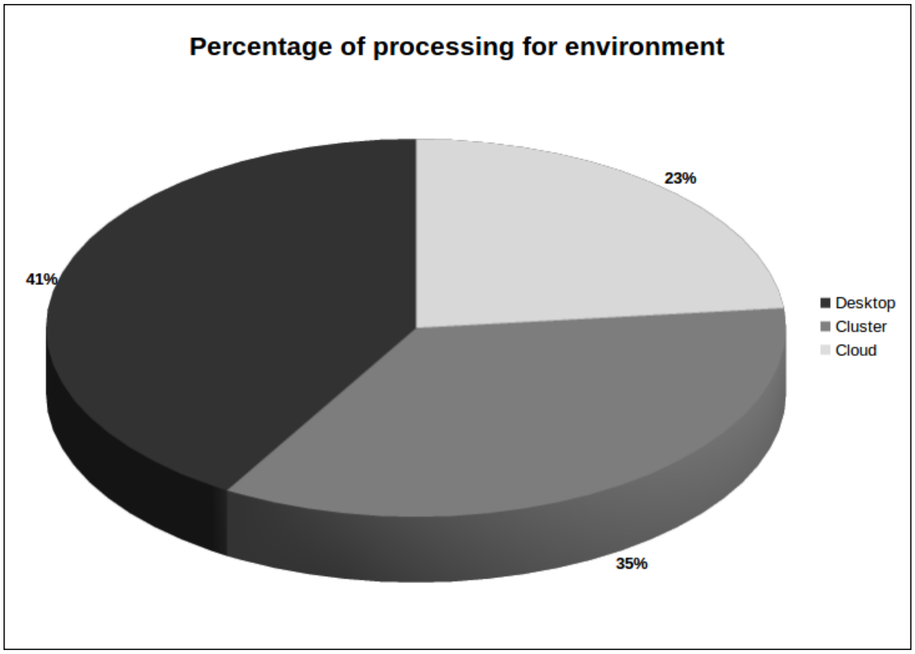


Figure 14: Division of jobs according to proteins workload for the corresponding machine capacity.

According to the results, best fits can be defined for experiments with no overload of a machine's capacity and the purchase of over qualified (and expensive) hardware is not necessary for the running of experiments. Moreover, investments in local hardware, either independent hosts or clusters, are advantageous for the development of research. Cloud computing has many conveniences, but does not work best to every case. We consider local management, setup freedom and avoidance of bandwidth limitations are a strong defense of the installation of local systems and enable remote access, when necessary. Paying for cloud solutions on a long term may be a wrong choice, as that requires high investments.

6. Related Work

Performance and user's experience have been addressed in scientific platforms, both on structural level and software implementations. Techniques that improve quality and reduce limitations involve diverse experimentation aspects.

Moreno et al. [49] developed a library for the processing of long protein sequences. The experiments revealed the bioinformatics bottleneck had changed from data acquisition to data interpretation.

Thaman and Singh [50] reviewed the services offered in distributed architectures and the task scheduling was identified as the most influential factor for the extraction of computational resources performance. Shagwan and Kumar [51] conducted an extensive review of scheduling algorithms for clouds. Both studies support our decision of applying machine learning on the Decision Maker.

Bianchi et al. [52] introduced a workflow management system to address Next Generation Sequencing (NGS). However, the authors did not consider cluster and cloud infrastructures for running the experiments and the solution is closed and not adaptable to other applications. Akos et al. [53] proposed a generic Science Gateway where applications can be uploaded as black box components. Althought it can be used as a basis for new portals, it is not an optimized solution and might lead to poor performance. Kacsuk [54] described a portal for the integration of tools that use grid systems for the execution of tasks. Although it is no longer available, it could be used for evaluation purposes.

Stitz et al. [55] proposed a solution to deal with the lack of patterns on virtualized nodes. It is a visualization system that supports the arrangement of the computational resources offered. However, the study did not address the methods appropriately used and failed to consider a variety of machines. Liu et al. [56] integrated Galaxy with Globus Transfer for a reliable data moving. The resources are provisioned on-demand, however, they cover exclusively paid cloud machines and do not fit any user. The issues addressed by such studies are common in distributed systems. Different technologies and strategies are tested to solve scalability, scheduling, monitoring, and heterogeneous platforms problems.

Sandes et al. [9] reviewed the architectures from FPGAs to GPU applied to bioinformatics and compared solutions that best fitted each architecture. The study showed the necessity of highly parallel solutions and platform variations. Mrozek et al. [57] introduced Cloud4PSP, a solution that adopts a scalability model to run *ab initio* proteins in clouds. It is based on the pay-as-you-go model and relies on horizontal and vertical scalability for enhancing the performance of the predictions. However, it is a closed solution strictly tied to a company.

Karoczkai et al. [58] presented a meta-broker for science gateways for scheduling jobs to heterogeneous machines. It creates a layer on the gUSE gateway to set job priorities and distributes them according to the distance from of the service providers. Although it can be a solution to load balancing, it does not fit the workload in available resources, which might impair its performance. Sandes et al. [59] filled this gap proposing an implementation that considers the user's expertise to provide wavefront balancing for a multinode arrangement. The solution maintains fair job shares, depending on the amount of data transfer and communication, and could be useful in applications that require less computational power.

Sandes et al. [19] proposed an extension of CUDAlign [31]. It consists of a multi-platform implementation for sequence alignment (MASA) that considers processing on GPU, FPGA, and CellBe architectures. It applies an optimization for reducing the number of cells calculated without losing precision and uses heterogenous environments; however, it addresses smaller workloads.

Macedo et al. [60] proposed an allocation policy based on master/slave strategy for heterogeneous multi-core clusters. Their study showed a proper placement of master nodes can reduce processing time. The experiments were limited by the cluster capacities and did not consider scaling down of the machines for smaller requirements.

Science gateways are multi-tenant systems that can benefit from specific provision policies. Peng et al. [61] described issues related to the maintenance of multi-tenant systems and introduced a knowledge-based resource allocation

Table 7: Evaluation of Related Works.

Work	**Platform**	**Scalable**	**Reproducibly**	**Flexible**	**Economy**
Akos et al. [53]	Grid	Yes	No	No	Yes
Liu et al. [56]	Cloud	Yes	Yes	No	No
Peng et al. [61]	Cloud	No	No	Yes	Yes
Ying and Lei [62]	Cloud	Yes	No	No	Yes
Bianchi et al. [52]	Cloud	No	Yes	No	No
Karoczkai et al. [58]	Multi	Yes	Yes	No	No
Sandes et al. [19]	Multi	No	No	Yes	Yes
Sandes et al. [9]	Multi	No	No	Yes	No
Garza et al. [18]	Multi	Yes	Yes	Yes	No

manager. The results showed reasonable execution time, however, the manager module produces additional costs.

Ying and Lei [62] designed a dynamic scheduler for multi-tenant systems that uses a mechanism based on preconditions for improving scheduling and reducing execution time. However, the approach must be tested in wider systems under harder conditions.

Garza et al. [18] introduced a set of tools for workflow scheduling. The idea is to distribute small workflow tasks to heterogeneous machines. It has opened up a new perspective for the design and execution of workflows, however, it does not consider the workload influence and its relation to the computing capacity.

This section has addressed the gaps on Scientific Gateways and integration research. Table 7 summarizes the aspects and differences among the related works and our novel solution for comparison purposes. Decision Maker is a solution based on execution history for properly modeling workload, according to users' claims and the literature. It suggests a scalable and flexible environment for workflow execution considering reproducibility and budget. To the best of our knowledge, Decision Maker is a good contribution to those systems and could be a powerful tool for improving the execution and provision of computational resources.

7. Conclusions

Scientific Gateways have become a handy tool to support researchers regarding experiment execution, data storage, and dissemination of results. However, in many cases systems show limitations, such as low processing capacity, storage space and response time. Such problems configure capacity bottlenecks, which compromise the efficiency of the systems.

Based on Galaxy experiments, we evaluated the retrieved information and extracted a general model from the data by using different regression techniques. We have proposed a decision module for improving the flexibility of execution environments and offering a smooth transition between multiple environments. The solution can also adapt to independent user's investment capabilities and avoid waste of resources. The module and its unique machine learning application is the differential component of infrastructure that supports current environments.

The SVR model showed high accuracy in predicting the proper machine to execute the workflow based on user's parameters definition and a good estimation of the processing time. A varied environment can be the best solution, once it includes cloud computing and offers other types of machines.

As future work, we intend to model an architecture on WorkflowSim for a complete view of the science gateway capacity, considering the whole network, connections and modules involved. Decision Maker is on the core of the architecture, towards aiding the definition of the best machine capacity to scientific experiments.

References

[1] E. Afgan, D. Baker, M. van den Beek, D. Blankenberg, D. Bouvier, M. Cech, J. Chilton, D. Clements, N. Coraor, C. Eberhard, B. Gruning, A. Guerler, J. Hillman-Jackson, G. Von Kuster, E. Rasche, N. Soranzo, N. Turaga, J. Taylor, A. Nekrutenko, J. Goecks, The galaxy platform for accessible, reproducible and collaborative biomedical analyses: 2016 update, Nucleic Acids Research 44 (W1) (2016) W3. `doi:10.1093/nar/gkw343`.
URL `http://dx.doi.org/10.1093/nar/gkw343`

[2] K. Wolstencroft, R. Haines, D. Fellows, A. Williams, D. Withers, S. Owen, S. Soiland-Reyes, I. Dunlop, A. Nenadic, P. Fisher, J. Bhagat, K. Belhajjame, F. Bacall, A. Hardisty, A. Nieva de la Hidalga, M. P. Balcazar Vargas, S. Sufi, C. Goble, The taverna workflow suite: designing and executing workflows of web services on the desktop, web or in the cloud, Nucleic Acids Research 41 (W1) (2013) W557. `arXiv:/oup/backfile/Content_public/Journal/nar/41/W1/10.1093/nar/gkt328/2/gkt328.pdf`, `doi:10.1093/nar/gkt328`.
URL `+http://dx.doi.org/10.1093/nar/gkt328`

[3] A. Balasko, Z. Farkas, P. Kacsuk, Building science gateways by utilizing the generic ws-pgrade/guse workflow system, Computer Science 14 (2) (2013) 307.
URL `https://journals.agh.edu.pl/csci/article/view/284`

[4] M. Reich, T. Liefeld, J. Gould, J. Lerner, P. Tamayo, J. P. Mesirov, GenePattern 2.0, Nat Genet 38 (5) (2006) 500–501. `doi:10.1038/ng0506-500`.
URL `http://dx.doi.org/10.1038/ng0506-500`

[5] C. A. Miller, Y. Qiao, T. DiSera, B. D'Astous, G. T. Marth, bam.iobio: a web-based, real-time, sequence alignment file inspector, Nat Meth 11 (12) (2014) 1189. `doi:10.1038/nmeth.3174`.
URL `http://dx.doi.org/10.1038/nmeth.3174`

[6] F. Chelaru, L. Smith, N. Goldstein, H. C. Bravo, Epiviz: interactive visual analytics for functional genomics data, Nature Methods 11 (9) (2014) 938–940. doi:10.1038/nmeth.3038.
URL http://dx.doi.org/10.1038/nmeth.3038

[7] M. Griffith, O. L. Griffith, S. M. Smith, A. Ramu, M. B. Callaway, A. M. Brummett, M. J. Kiwala, A. C. Coffman, A. A. Regier, B. J. Oberkfell, G. E. Sanderson, T. P. Mooney, N. G. Nutter, E. A. Belter, F. Du, R. L. Long, T. E. Abbott, I. T. Ferguson, D. L. Morton, M. M. Burnett, J. V. Weible, J. B. Peck, A. Dukes, J. F. McMichael, J. T. Lolofie, B. R. Derickson, J. Hundal, Z. L. Skidmore, B. J. Ainscough, N. D. Dees, W. S. Schierding, C. Kandoth, K. H. Kim, C. Lu, C. C. Harris, N. Maher, C. A. Maher, V. J. Magrini, B. S. Abbott, K. Chen, E. Clark, I. Das, X. Fan, A. E. Hawkins, T. G. Hepler, T. N. Wylie, S. M. Leonard, W. E. Schroeder, X. Shi, L. K. Carmichael, M. R. Weil, R. W. Wohlstadter, G. Stiehr, M. D. McLellan, C. S. Pohl, C. A. Miller, D. C. Koboldt, J. R. Walker, J. M. Eldred, D. E. Larson, D. J. Dooling, L. Ding, E. R. Mardis, R. K. Wilson, Genome modeling system: A knowledge management platform for genomics, PLOS Computational Biology 11 (7) (2015) 1–21. doi:10.1371/journal.pcbi.1004274.
URL https://doi.org/10.1371/journal.pcbi.1004274

[8] J. Severin, M. Lizio, J. Harshbarger, H. Kawaji, C. O. Daub, Y. Hayashizaki, T. F. Consortium, N. Bertin, A. R. R. Forrest, Interactive visualization and analysis of large-scale sequencing datasets using zenbu, Nat Biotech 32 (3) (2014) 217–219. doi:10.1038/nbt.2840.
URL http://dx.doi.org/10.1038/nbt.2840

[9] E. F. D. O. Sandes, A. Boukerche, A. C. M. A. D. Melo, Parallel optimal pairwise biological sequence comparison: Algorithms, platforms, and classification, ACM Comput. Surv. 48 (4) (2016) 63:1–63:36. doi:

10.1145/2893488.
URL http://doi.acm.org/10.1145/2893488

[10] S. Gesing, J. Krüger, R. Grunzke, S. Herres-Pawlis, A. Hoffmann, Using science gateways for bridging the differences between research infrastructures, Journal of Grid Computing 14 (4) (2016) 545–557. doi:10.1007/s10723-016-9385-8.
URL http://dx.doi.org/10.1007/s10723-016-9385-8

[11] R. A. Faccioli, Implementao de um Framework de Computao Evolutiva Multi-Objetivo para Predio Ab Initio da Estrutura Terciria de Protenas, "http://www.teses.usp.br/teses/disponiveis/18/18153/tde-09052013-145839/pt-br.php" (2012).

[12] L. Zimmerman, R. Grunzke, J. Kruguer, Maintaining a science gateway - lessons learned from mosgrid, in: International Conference on System Sciences, Hawaii, EUA, 2017, p. 10.

[13] F. Jrad, J. Tao, A. Streit, A broker-based framework for multi-cloud workflows, in: Proceedings of the 2013 International Workshop on Multi-cloud Applications and Federated Clouds, MultiCloud '13, ACM, New York, NY, USA, 2013, pp. 61–68. doi:10.1145/2462326.2462339.
URL http://doi.acm.org/10.1145/2462326.2462339

[14] K. Brown, K. Grolinger, M. Capretz, Data providing web service-based integration framework for use in a health care context, in: Electrical and Computer Engineering (CCECE), 2011 24th Canadian Conference on, 2011, pp. 001069–001072. doi:10.1109/CCECE.2011.6030625.

[15] J. Li, B. Song, Web services integration on data mining based on soa, in: Intelligence Information Processing and Trusted Computing (IPTC), 2010 International Symposium on, 2010, pp. 532–534. doi:10.1109/IPTC.2010.147.

[16] M. Papazoglou, Service-oriented computing: concepts, characteristics and directions, in: Web Information Systems Engineering, 2003. WISE 2003. Proceedings of the Fourth International Conference on, 2003, pp. 3–12. doi:10.1109/WISE.2003.1254461.

[17] J. Kovcs, P. Kacsuk, A. Lomaka, Using a private desktop grid system for accelerating drug discovery, Future Generation Computer Systems 27 (6) (2011) 657 – 666. doi:https://doi.org/10.1016/j.future.2010.12.008.
URL http://www.sciencedirect.com/science/article/pii/S0167739X10002578

[18] L. de la Garza, J. Veit, A. Szolek, M. Röttig, S. Aiche, S. Gesing, K. Reinert, O. Kohlbacher, From the desktop to the grid: scalable bioinformatics via workflow conversion, BMC Bioinformatics 17 (1) (2016) 127. doi:10.1186/s12859-016-0978-9.
URL http://dx.doi.org/10.1186/s12859-016-0978-9

[19] E. F. De O. Sandes, G. Miranda, X. Martorell, E. Ayguade, G. Teodoro, A. C. M. A. De Melo, Masa: A multiplatform architecture for sequence aligners with block pruning, ACM Trans. Parallel Comput. 2 (4) (2016) 28:1–28:31. doi:10.1145/2858656.
URL http://doi.acm.org/10.1145/2858656

[20] B. Abdul-Wahid, L. Yu, D. Rajan, H. Feng, E. Darve, D. Thain, J. Izaguirre, Folding proteins at 500 ns/hour with work queue, in: E-Science (e-Science), 2012 IEEE 8th International Conference on, 2012, pp. 1–8. doi:10.1109/eScience.2012.6404429.

[21] S. Pronk, G. R. Bowman, B. Hess, P. Larsson, I. S. Haque, V. S. Pande, I. Pouya, K. Beauchamp, P. M. Kasson, E. Lindahl, Copernicus: A new paradigm for parallel adaptive molecular dynamics, in: 2011 International Conference for High Performance Computing, Networking, Storage and Analysis (SC), 2011, pp. 1–10. doi:10.1145/2063384.2063465.

[22] E. Alm, D. Baker, Prediction of protein-folding mechanisms from free-energy landscapes derived from native structures, Journal of Molecular Biology 96 (1999) 1130511310.

[23] M. T. Hoque, M. Chetty, L. S. Dooley, A guided genetic algorithm for protein folding prediction using 3d hydrophobic-hydrophilic model, in: 2006 IEEE International Conference on Evolutionary Computation, 2006, pp. 2339–2346. doi:10.1109/CEC.2006.1688597.

[24] A. Lehninger, D. L. Nelson, M. M. Cox, Lehninger Principles of Biochemistry, fifth edition Edition, W. H. Freeman, 2008.
URL http://www.amazon.ca/exec/obidos/redirect?tag=citeulike09-20%255C&amp;path=ASIN/1429224169

[25] S. Chatterjee, R. A. Smrity, M. R. Islam, Protein structure prediction using chemical reaction optimization, in: 2016 19th International Conference on Computer and Information Technology (ICCIT), 2016, pp. 321–326. doi:10.1109/ICCITECHN.2016.7860217.

[26] C. R. S. Brasil, A. C. B. Delbem, F. L. B. da Silva, Multiobjective evolutionary algorithm with many tables for purely ab initio protein structure prediction, Journal of Computational Chemistry 34 (20) (2013) 1719–1734. doi:10.1002/jcc.23315.

[27] S. Thomas, N. Amato, Parallel protein folding with stapl, in: Parallel and Distributed Processing Symposium, 2004. Proceedings. 18th International, 2004, pp. 189–. doi:10.1109/IPDPS.2004.1303204.

[28] N. Phuoc, S.-R. Kim, Protein fold prediction using cluster merging, in: Computer Sciences and Convergence Information Technology (ICCIT), 2011 6th International Conference on, 2011, pp. 293–298.

[29] I. Sovic, N. Antulov-Fantulin, I. Canadi, M. Piskorec, M. Siki, Parallel protein docking tool, in: MIPRO, 2010 Proceedings of the 33rd International Convention, 2010, pp. 1333–1338.

[30] A. Sharma, A. Papanikolaou, E. Manolakos, Accelerating all-to-all protein structures comparison with tmalign using a noc many-cores processor architecture, in: Parallel and Distributed Processing Symposium Workshops PhD Forum (IPDPSW), 2013 IEEE 27th International, 2013, pp. 510–519. doi:10.1109/IPDPSW.2013.222.

[31] E. F. d. O. Sandes, G. Miranda, X. Martorell, E. Ayguade, G. Teodoro, A. C. M. Melo, Cudalign 4.0: Incremental speculative traceback for exact chromosome-wide alignment in gpu clusters, IEEE Transactions on Parallel and Distributed Systems 27 (10) (2016) 2838–2850. doi:10.1109/TPDS.2016.2515597.

[32] S. Vijayakumar, P. Lakshmi, A fuzzy inference system for predicting allergenicity and allergic cross-reactivity in proteins, in: Bioinformatics and Biomedicine (BIBM), 2013 IEEE International Conference on, 2013, pp. 49–52. doi:10.1109/BIBM.2013.6732458.

[33] K. Kavitha, R. Saritha, C. Vinod, Computational prediction of continuous b-cell epitopes using random forest classifier, in: Computing, Communications and Networking Technologies (ICCCNT),2013 Fourth International Conference on, 2013, pp. 1–5. doi:10.1109/ICCCNT.2013.6726820.

[34] K. Okada, L. Flores, M. Wong, D. Petkovic, Microenvironment-based protein function analysis by random forest, in: Pattern Recognition (ICPR), 2014 22nd International Conference on, 2014, pp. 3138–3143. doi:10.1109/ICPR.2014.541.

[35] D. de Lucena, T. Woerle de Lima, A. da Silva Soares, A. Delbem, A. Rodrigues Galvao Filho, C. Coelho, G. Laureano, Multi-objective evolutionary algorithm for variable selection in calibration problems: A case study for protein concentration prediction, in: Evolutionary Computation (CEC), 2013 IEEE Congress on, 2013, pp. 1053–1059. doi:10.1109/CEC.2013.6557683.

[36] J.-S. Yeh, D.-Y. Chen, M. Ouhyoung, A web-based protein retrieval system by matching visual similarity, in: Emerging Information Technology Conference, 2005., 2005, pp. 108–110. doi:10.1109/EITC.2005.1544360.

[37] K. Taha, P. Yoo, M. Al Zaabi, ipfpi: A system for improving protein function prediction through cumulative iterations, Computational Biology and Bioinformatics, IEEE/ACM Transactions on PP (99) (2014) 1–1. doi:10.1109/TCBB.2014.2344681.

[38] A. Mandal, I. Das, D. Bhattacharjee, A software tool for extraction of annotation data from a pdb file, in: Emerging Trends and Applications in Computer Science (NCETACS), 2012 3rd National Conference on, 2012, pp. 31–37. doi:10.1109/NCETACS.2012.6203293.

[39] J. Yang, Y. Zhang, I-tasser server: new development for protein structure and function predictions, Nucleic Acids Res 43 (Web Server issue) (2015) W174–W181, 25883148[pmid]. doi:10.1093/nar/gkv342.
URL http://www.ncbi.nlm.nih.gov/pmc/articles/PMC4489253/

[40] E. Afgan, D. Baker, M. van den Beek, D. Blankenberg, D. Bouvier, M. ech, J. Chilton, D. Clements, N. Coraor, C. Eberhard, B. Grning, A. Guerler, J. Hillman-Jackson, G. Von Kuster, E. Rasche, N. Soranzo, N. Turaga, J. Taylor, A. Nekrutenko, J. Goecks, The galaxy platform for accessible, reproducible and collaborative biomedical analyses: 2016 update, Nucleic Acids Research 44 (W1) (2016) W3, tESTE. arXiv:/oup/backfile/Content_public/Journal/nar/44/W1/10.1093_nar_gkw343/3/gkw343.pdf, doi:10.1093/nar/gkw343.
URL +http://dx.doi.org/10.1093/nar/gkw343

[41] B. Giardine, C. Riemer, R. C. Hardison, R. Burhans, L. Elnitski, P. Shah, Y. Zhang, D. Blankenberg, I. Albert, J. Taylor, W. Miller, W. J. Kent, A. Nekrutenko, Galaxy: A platform for interactive large-scale genome analysis, Genome Research 15 (10) (2005) 1451–1455. arXiv:http://genome.cshlp.org/content/15/10/1451.full.pdf+html, doi:

10.1101/gr.4086505.
URL http://genome.cshlp.org/content/15/10/1451.abstract

[42] A. Defelicibus, Koala: sistema para integração de métodos de predição e análise de estruturas de proteína, Disponível em: http://www.teses.usp.br/teses/disponiveis/82/82131/tde-22062016-102823., dissertação de Mestrado (2016).

[43] C. C. Aggarwal, Data mining: the textbook, Springer, 2015.

[44] A. J. Smola, B. Schölkopf, A tutorial on support vector regression, Statistics and computing 14 (3) (2004) 199–222.

[45] T. M. Mitchell, Machine Learning, 1st Edition, McGraw-Hill, Inc., New York, NY, USA, 1997.

[46] X. Yan, X. G. Su, Linear Regression Analysis: Theory and Computing, World Scientific Publishing Co., Inc., River Edge, NJ, USA, 2009.

[47] R. Tibshirani, Regression shrinkage and selection via the lasso, Journal of the Royal Statistical Society, Series B 58 (1994) 267–288.

[48] A. E. Hoerl, R. W. Kennard, Ridge regression: Biased estimation for nonorthogonal problems, Technometrics 42 (1) (2000) 80–86. doi:10.2307/1271436.
URL http://dx.doi.org.ez67.periodicos.capes.gov.br/10.2307/1271436

[49] A. R. Moreno, Ó. T. Tirado, O. T. Salazar, Out of Core Computation of HSPs for Large Biological Sequences, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013, Ch. 22, pp. 189–199. doi:10.1007/978-3-642-38682-4_22.
URL http://dx.doi.org/10.1007/978-3-642-38682-4_22

[50] J. Thaman, M. Singh, Current perspective in task scheduling techniques in cloud computing : A review, International Journal in Foundations of Com-

puter Sc ience & Technology (IJFCST) 6 (1). doi:DOI:10.5121/ijfcst.2016.6106.

[51] J. Bhagwan, S. Kumar, An intense review of task scheduling algorithms in cloud computing, International Journal of Advanced Research in Computer and Communication Engineering 5 (11). doi:DOI10.17148/IJARCCE.2016.511128.

[52] V. Bianchi, A. Ceol, A. G. Ogier, S. de Pretis, E. Galeota, K. Kishore, P. Bora, O. Croci, S. Campaner, B. Amati, M. J. Morelli, M. Pelizzola, Integrated Systems for NGS Data Management and Analysis: Open Issues and Available Solutions, Front Genet 7 (2016) 75.

[53] A. Balasko, Z. Farkas, P. Kacsuk, Building science gateways by utilizing the generic ws-pgrade/guse workflow system, Computer Science 14 (2) (2013) 307. doi:http://dx.doi.org/10.7494/csci.2013.14.2.307.
URL https://journals.agh.edu.pl/csci/article/view/284

[54] P. Kacsuk, P-grade portal family for grid infrastructures, Concurr. Comput. : Pract. Exper. 23 (3) (2011) 235–245. doi:10.1002/cpe.1654.
URL http://dx.doi.org.ez67.periodicos.capes.gov.br/10.1002/cpe.1654

[55] H. Stitz, S. Gratzl, M. Krieger, M. Streit, Cloudgazer: A divide-and-conquer approach to monitoring and optimizing cloud-based networks, in: 2015 IEEE Pacific Visualization Symposium (PacificVis), 2015, pp. 175–182. doi:10.1109/PACIFICVIS.2015.7156375.

[56] B. Liu, R. K. Madduri, B. Sotomayor, K. Chard, L. Lacinski, U. J. Dave, J. Li, C. Liu, I. T. Foster, Cloud-based bioinformatics workflow platform for large-scale next-generation sequencing analyses, Journal of Biomedical Informatics 49 (2014) 119 – 133. doi:http://dx.doi.org/10.1016/j.jbi.2014.01.005.
URL http://www.sciencedirect.com/science/article/pii/S1532046414000070

[57] D. Mrozek, P. Gosk, B. Małysiak-Mrozek, Scaling ab initio predictions of 3d protein structures in microsoft azure cloud, Journal of Grid Computing 13 (4) (2015) 561–585. doi:10.1007/s10723-015-9353-8.
URL http://dx.doi.org/10.1007/s10723-015-9353-8

[58] K. Karoczkai, A. Kertesz, P. Kacsuk, A meta-brokering framework for science gateways, Journal of Grid Computing 14 (4) (2016) 687–703. doi:10.1007/s10723-016-9378-7.
URL https://doi.org/10.1007/s10723-016-9378-7

[59] E. F. de O. Sandes, C. G. Ralha, A. C. M. de Melo, An agent-based solution for dynamic multi-node wavefront balancing in biological sequence comparison, Expert Systems with Applications 41 (10) (2014) 4929 – 4938. doi:http://dx.doi.org/10.1016/j.eswa.2014.01.030.
URL http://www.sciencedirect.com/science/article/pii/S0957417414000542

[60] E. de Araujo Macedo, A. C. Magalhaes Alves de Melo, G. H. Pfitscher, A. Boukerche, Multiple biological sequence alignment in heterogeneous multicore clusters with user-selectable task allocation policies, The Journal of Supercomputing 63 (3) (2013) 740–756. doi:10.1007/s11227-012-0768-8.
URL http://dx.doi.org/10.1007/s11227-012-0768-8

[61] G. Peng, H. Wang, H. Zhang, J. Dong, Knowledge-based resource allocation for collaborative simulation development in a multi-tenant cloud computing environment, IEEE Transactions on Services Computingdoi:10.1109/TSC.2016.2518161.

[62] F. Ying, G. Lei, Optimal scheduling simulation of software for multi-tenant in cloud computing environment, in: 2014 Fifth International Conference on Intelligent Systems Design and Engineering Applications, 2014, pp. 688–692. doi:10.1109/ISDEA.2014.158.